

IN THE CLAIMS:

Please cancel claims 2-6, 15, and 34-36 without prejudice.

1 1. (Previously Presented) A method for converting a file access data structure from a
2 first endianness to a second endianness by a processor, the method comprising the steps
3 of:

4 identifying, from a descriptor look up table, a series of actions to perform on
5 elements of the file access data structure, where the series of actions include at least one
6 of converting, copying, or linking; and

7 performing the identified series of actions on the elements of the file access data
8 structure to convert the file data structure from the first endianness to the second
9 endianness.

2.-6. (Cancelled)

7.-14. (Cancelled)

15. (Cancelled)

16.-19. (Cancelled)

1 20. (Previously Presented) A method for converting a data structure by a processor,
2 comprising:

3 calling a byte-swapping engine;

4 providing a file access data structure as input to the byte-swapping engine;

5 providing a descriptor look up table to the byte-swapping engine;

6 identifying, from the descriptor look up table, a series of actions to perform on
7 elements of the file access data structure in order to swap bytes of the file access data

8 structure from a first endianness to a second endianness, where the series of actions
9 include at least one of converting, copying, or linking; and
10 performing the identified series of actions on the elements of the file access data
11 structure to convert the file access data structure.

1 21. (Previously Presented) The method as in claim 20, further comprising:
2 using as the file access data structure a file having Direct Access File System
3 (DAFS) protocol.

1 22. (Previously Presented) The method as in claim 20, further comprising:
2 determining if the file access data structure is a critical path data structure, where
3 the critical path data structure includes commonly utilized data structures, and if the file
4 access data structure is a critical path data structure, perform byte swap operations using
5 specific code functions.

1 23. (Previously Presented) The method as in claim 20, further comprising:
2 determining if the file access data structure is a critical path data structure, where
3 the critical path data structure includes commonly utilized data structures, and if the file
4 access data structure is not a critical path data structure, perform byte swap operations on
5 a data structure header.

1 24. (Previously Presented) The method as in claim 20, further comprising:
2 swapping bytes of the data structure as needed, in response to swapping bytes of
3 the file access data structure.

1 25. (Previously Presented) The method as in claim 20, further comprising:
2 determining if an element entry of the descriptor look up table is nested;
3 branching to the nested entry;
4 identifying, from the descriptor look up table, a nested series of actions to perform
5 on elements of the nested entry in order to swap bytes of the entry from a first endianness

6 to a second endianness, where the nested series of actions includes linking and
7 converting.

- 1 26. (Previously Presented) A computer to convert a data structure by a processor,
2 comprising:
3 means for calling a byte-swapping engine;
4 means for providing a file access data structure as input to the byte-swapping
5 engine;
6 means for providing a descriptor look up table to the byte-swapping engine;
7 means for identifying, from the descriptor look up table, a series of actions to
8 perform on elements of the file access data structure in order to swap bytes of the file
9 access data structure from a first endianness to a second endianness, where the series of
10 actions include at least one of converting, copying, or linking; and
11 means for performing the identified series of actions on the elements of the file
12 access data structure to convert the file access data structure.
- 1 27. (Previously Presented) The computer as in claim 26, further comprising:
2 means for using as the file access data structure a file having Direct Access File
3 System (DAFS) protocol.
- 1 28. (Previously Presented) The computer as in claim 26, further comprising:
2 means for determining if the file access data structure is a critical path data
3 structure, where the critical path data structure includes commonly utilized data
4 structures, and if the file access data structure is a critical path data structure, perform
5 byte swap operations using specific code functions.
- 1 29. (Previously Presented) The computer as in claim 26, further comprising:
2 means for determining if the file access data structure is a critical path data
3 structure, where the critical path data structure includes commonly utilized data
4 structures, and if the file access data structure is not a critical path data structure, perform
5 byte swap operations on a data structure header.

- 1 30. (Previously Presented) The computer as in claim 26, further comprising:
 - 2 means for swapping bytes of the data structure as needed, in response to swapping
 - 3 bytes of the file access data structure.
- 1 31. (Previously Presented) The computer as in claim 26, further comprising:
 - 2 means for determining if an element entry of the descriptor look up table is
 - 3 nested;
 - 4 means for branching to the nested entry;
 - 5 means for identifying, from the descriptor look up table, a nested series of actions
 - 6 to perform on elements of the nested entry in order to swap bytes of the entry from a first
 - 7 endianness to a second endianness, where the nested series of actions includes converting
 - 8 and linking.
- 1 32. (Previously Presented) A computer readable media, comprising:
 - 2 said computer readable media containing instructions for execution on a processor
 - 3 for the practice of a method for converting a data structure by a processor, the method
 - 4 having the steps of,
 - 5 calling a byte-swapping engine;
 - 6 providing a file access data structure as input to the byte-swapping engine;
 - 7 providing a descriptor look up table to the byte-swapping engine;
 - 8 identifying, from the descriptor look up table, a series of actions to perform on
 - 9 elements of the file access data structure in order to swap bytes of the file access data
 - 10 structure from a first endianness to a second endianness, where the series of actions
 - 11 include at least one of converting, copying, or linking; and
 - 12 performing the identified series of actions on the elements of the file access data
 - 13 structure to convert the file access data structure.
- 1 33. (Cancelled)
- 1 34.-36. (Cancelled)

- 1 37. (Previously Presented) A method for converting a first data structure from a to a
2 second data structure by a processor, the method comprising the steps of:
 - 3 using a descriptor lookup table to provide actions to be performed on each
4 element of the first data structure; and
 - 5 stepping through the descriptor table and processing each element of the first data
6 structure according to the element's size and action to convert the first data structure into
7 the second data structure.

- 1 38. (Previously Presented) The method of claim 37, further comprising:
 - 2 using a byte as the data structure.

- 1 39. (Previously Presented) The method of claim 2, wherein the critical data path structure
2 includes commonly used data structures.

- 1 40. (Previously Presented) The method of claim 2, wherein the critical data path structure
2 is a direct access file system (DAFS) header data structure.

- 1 41. (Previously Presented) The method of claim 2, wherein the specific code functions
2 are designed to rapidly convert any elements of the data structure to the second
3 endianness without using a byte swapping engine.

- 1 42. (Previously Presented) The computer-readable medium of claim 15, wherein the
2 critical data path structure includes commonly used data structures.

- 1 43. (Previously Presented) The computer-readable medium of claim 15, wherein the
2 critical data path structure is a direct access file system (DAFS) header data structure.

- 1 44. (Previously Presented) The computer-readable medium of claim 15, wherein the
2 specific code functions are designed to rapidly convert any elements of the data structure
3 to the second endianness without using a byte swapping engine.

1 45. (Previously Presented) The method of claim 34, wherein the critical data path
2 structure includes commonly used data structures.

1 46. (Previously Presented) The method of claim 34, wherein the critical data path
2 structure is a direct access file system (DAFS) header data structure.

1 47. (Previously Presented) The method of claim 34, wherein the specific code
2 functions are designed to rapidly convert any elements of the data structure to the
3 second endianness without using a byte swapping engine.